

(h)

HDL INPUT METHOD AND DEVICE THEREFOR

Patent Number: JP2000132584
Publication date: 2000-05-12
Inventor(s): HARA NAOKI; MORIWAKI IKU
Applicant(s):: HITACHI LTD
Requested Patent: ☐ JP2000132584 (JP00132584)
Application Number: JP19980304994 19981027
Priority Number(s):
IPC Classification: G06F17/50
EC Classification:
Equivalents:

Abstract

PROBLEM TO BE SOLVED: To provide an HDL description input system reducing the burden on a designer for the HDL(hardware description language) description and preventing a grammar error of the designer.

SOLUTION: An HDL syntax data base 104a having the models of respective HDL syntaxes and an associated design data base 104b having connection information between circuits are installed. An HDL syntax generation part 100c generating the HDL syntax from the data base 104a with the reservation word of HDL which a designer inputs as a key, an associated description generation part 100d generating the HDL syntax from the model of the HDL syntax and circuit information when circuit information required for input exists in the data base 104b, an associated description correction part 100e correcting an affected HDL text file from the data base 104b when the correction of the generated HDL text file 104c affects the other HDL text file and a semantics check part 100f checking semantics having no problem in terms of a circuit, are installed.

Data supplied from the esp@cenet database - I2

(h)

(19) 日本国特許庁 (JP)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開 2000-132584

(P2000-132584A)

(43) 公開日 平成12年5月12日 (2000. 5. 12)

(51) Int. Cl. 7

識別記号

F I

テーマコード (参考)

G 0 6 F 17/50

G 0 6 F 15/60

6 5 4 R 5B046

6 6 4 Z

審査請求 未請求 請求項の数 5

〇 L

(全 12 頁)

(21) 出願番号

特願平10-304994

(22) 出願日

平成10年10月27日 (1998. 10. 27)

請求項5

(71) 出願人 000005108

株式会社日立製作所

東京都千代田区神田駿河台四丁目6番地

(72) 発明者 原 直樹

神奈川県秦野市堀山下1番地 株式会社日

立インフォメーションテクノロジー内

(72) 発明者 森脇 郁

神奈川県川崎市幸区鹿島田890番地 株式

会社日立製作所システム開発本部内

(74) 代理人 100068504

弁理士 小川 勝男

Fターム(参考) 5B046 AA08 BA03 JA01 KA03 KA05

(54) 【発明の名称】 HDL入力方法及び装置

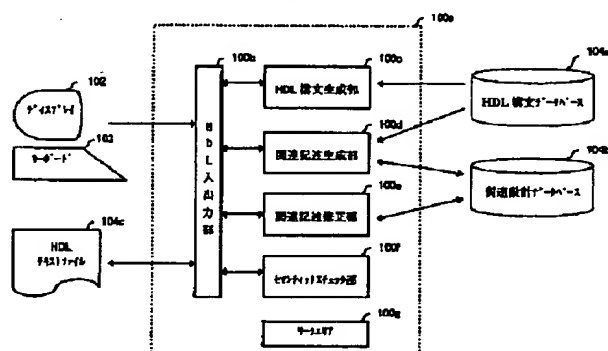
(57) 【要約】

(修正有)

【課題】設計者のHDL (ハードウェア記述言語) 記述の負担の低減と文法ミスを防ぐHDL記述入力システムを提供する。

【解決手段】各HDL構文の雛形を持ったHDL構文データベース104aと、各回路間の接続情報を持った関連設計データベース104bを備え、設計者が入力したHDLの予約語を鍵にデータベース104aからHDL構文を生成するHDL構文生成部100cと、データベース104bに入力に必要な回路情報がある場合にHDL構文の雛形と回路情報からHDL構文を生成する関連記述生成部100dと、既作成済みのHDLテキストファイル104cの修正が他のHDLテキストファイルに影響する場合、データベース104bから影響するHDLテキストファイルの修正を行う関連記述修正部100eと、回路的に問題がないセマンティクスチェックを行うセマンティクスチェック部100fを備える。

図 2



【特許請求の範囲】

【請求項 1】ハードウェア記述言語（以下HDLと言う）を用いて回路設計を行うためのHDL入力装置において、各HDLの構文の雛形を持ったデータベースと、各回路間の接続情報を持ったデータベースを参照する手段を備え、設計者が入力したHDLの予約語からHDL構文を生成するプログラムと、回路間の関連情報を持ったデータベースから回路情報を生成しHDL構文を生成するプログラムと、既存のHDLテキストファイルの変更が他のHDLテキストファイルに関連している場合関連箇所を修正するプログラムと、回路のセマンティクスチェックを行うプログラムを備えたHDL入力装置。

【請求項 2】前記請求項 1 のHDL入力装置において、設計者が入力したHDLの予約語を鍵に、HDL構文の雛形を持つデータベースを検索し、雛形から予約語のHDL構文を生成するプログラム。

【請求項 3】前記請求項 1 のHDL入力装置において、回路間の接続情報を持つデータベースに入力に必要な回路情報が登録されている場合に、HDL構文の雛形と回路情報からHDL構文を生成するプログラム。

【請求項 4】前記請求項 1 のHDL入力装置において、機能変更や論理不良等により既作成済みのHDLテキストファイルの変更を行うことで他のHDLテキストファイルも変更が必要ないか回路間の接続情報を持つデータベースを検索し変更が必要な場合他のHDLテキストファイルも変更を行うプログラム。

【請求項 5】前記請求項 1 のHDL記述入力装置において、未使用な信号を記述や、機能レベル／ゲートレベルのシミュレーション結果に影響があるprocess文、always文のセンシティブリティリストのセマンティクスチェックを行うプログラム。

【発明の詳細な説明】**【0001】**

【発明の属する技術分野】本発明は、HDLを用いた回路設計の開発に関するものである。

【0002】

【従来の技術】半導体製造技術の進展により1チップ上に搭載できるゲート数は増加し続けている。こうした状況からこれまでのゲートレベルの回路図入力による設計手法に代わって、ゲートレベルより上位の機能レベルを表現可能なHDLによる設計手法に移行している。HDL設計手法では高度な記述が可能な反面、設計者がHDL文法を学習し、シンタックスに注意しながら機能を満たす回路記述の入力を行わなければならないという複雑さがある。

【0003】上記の如き問題を解決する一手法として、日本特許特開平6-314314では回路の機能と生成される回路の目標値を与えてHDLを作成する入力装置が示されている。

【0004】

【発明が解決しようとする課題】HDLを用いた回路設計では、設計者がHDL文法を学習し入力を行わなければならない。

【0005】そのためHDL文法のシンタックスに注意しながら入力をするだけでなく、HDL設計の経験が少ないと記述ミスを起こし易い。

【0006】また論理不良や仕様変更による記述に変更が発生し、その変更が関連して他の記述箇所も修正が必要な場合、HDL設計に熟練した設計者でも見逃す可能性がある。

【0007】日本特許特開平6-314314では、記述の自由度が小さく、論理変更等による関連箇所の記述の変更に対応していない。

【0008】本発明は、前記の従来の問題点を解決し設計者の負担を減らしたHDL入力システムである。

【0009】

【課題を解決するための手段】上記課題を解決するため、本発明では、各HDL構文の雛形を持ったデータベースと、各回路間の接続情報を持ったデータベースを参照する手段を備え、設計者が入力したHDLの予約語を鍵にHDL構文の雛形情報を持ったデータベースからHDL構文を生成する手段と、回路間の接続情報を持つデータベースに入力に必要な回路情報がある場合にHDL構文の雛形と回路情報からHDL構文を生成する手段と、既作成済みのHDLテキストファイルの修正が他のHDLテキストファイルに影響する場合、回路間の接続情報を持つデータベースから影響するHDLテキストファイルの修正を行う手段と、回路的に問題がないかセマンティクスチェックを行う手段を提供する。

【0010】

【発明の実施の形態】以下、本発明の実施形態を図を用いて説明する。

【0011】＜図1の説明＞図1は、HDL記述入力装置の構成図である。

【0012】HDL入力プログラム100aは主記憶装置100に格納される。主記憶装置100に格納したHDL入力プログラム100aを処理するCPU101、HDL入力プログラムの入力画面を表示するディスプレイ102、設計者がHDLの入力を行うためのキーボード103、HDLの予約語からHDL構文を生成するための雛形情報を持つHDL構文データベース104aと、機能変更や論理不良等で既作成済みのHDLテキストファイル104cの修正により他のHDLテキストファイル104cも修正する必要がある場合、その他のHDLテキストファイル104cの修正箇所を検出しHDLを修正するための回路の入出力端子情報と関連回路名称情報を持つ関連設計データベース104bとHDL入力プログラム100aから生成したHDLテキストファイル104cを保存するハードディスク104から構成され、それぞれの装置はバス105で接続されている。

【0013】設計者（操作者）はキーボード103からHDLの予約語や回路情報の入力を行う。ディスプレイ102は設計者が入力したHDLの文字列（予約語や論理記述）がHDL入力プログラム100aの入力画面に表示される。HDL入力プログラム100aはキーボード103から入力された文字列を入力データとしCPU101でHDL入力プログラム100aの命令が実行される。命令によりハードディスク104に格納されているHDL構文データベース104aや関連設計データベース104bの情報をもとにHDL構文の生成を行う。

【0014】設計者がHDL記述が完成したと判断したら設計者はHDL入力プログラム100aにHDLテキストファイル104cの出力指示を行いハードディスク104に出力する。

【0015】＜図2の説明＞図2は、HDL入力プログラム100aの構成図である。

【0016】HDL入出力部100bはキーボード103から入力したHDLの予約語である文字列データを受け取る。次に文字列データを各処理部に渡しHDL入力プログラム100aの制御を行う。またHDL入力プログラム100aの処理によって作成されたHDLテキストファイル104cをハードディスク104に出力する。

【0017】HDL構文生成部100cの入力データはHDL入出力部100bから受け取った文字列データ（予約語）である。受け取った文字列データを鍵にHDL構文データベース104aを参照し、該当する文字列データの雛形情報からHDL構文を生成しHDL入出力部100bに返す。

【0018】関連記述生成部100dはHDL入出力部100bから階層記述に関する文字列データ（予約語と回路名称）を受け取る。受け取った文字列データ（回路名称）を鍵に関連設計データベース104bを検索して文字列データ（回路名称）があれば関連設計データベース104bからHDL構文に必要な情報を作成し、また文字列データ（予約語）を鍵にHDL構文データベース104aからHDL構文の雛形情報を抽出し関連設計データベース104bとマージしてHDL構文を生成しHDL入出力部100bに返す。

【0019】関連記述修正部100eは既に作成済みのHDLテキストファイル104cを修正する場合に処理を行う。既に作成したHDLテキストファイル104cの入出力端子記述に対して端子名の変更や端子の追加または削除を行う場合に、HDL入出力部100bから回路名称と端子情報を受け取る。その端子が変更したHDLテキストファイル104c以外に記述されていないか端子を変更をした回路名称を鍵に関連設計データベース104bを検索する。検索の結果他のHDLテキストファイル104cで使用していたら変更した端子と同じ様に記述の変更を行いHDL入出力部100bに返す。

【0020】セマンティックチェック部100fはH

D/L入出力部100bから設計者が入力した信号宣言・入出力端子情報の文字列データを入力データとする。信号宣言・入出力端子情報と論理記述部の信号端子情報のそれぞれをHDL入力プログラム100aのワークエリア100gに格納する。そしてそれぞれの信号・端子情報を比較し過不足な信号・端子の記述がないかチェックを行い過不足があった場合はメッセージの出力を行う。

【0021】＜図3の説明＞図3は、HDL構文データベース104aのデータ構造を示す。

10 【0022】HDL構文データベース104aは、HDL構文生成部100c・関連記述生成部100dでHDL構文を生成するために使用する。HDL構文データベース104aは、設計者が入力したHDLの文字列（予約語）を鍵にして、HDL構文データベース104aの予約語302の検索を行う。設計者が入力した文字列と一致する予約語302があれば、予約語に対応する構文情報（雛形）303からHDL構文を生成する。

20 【0023】HDL構文データベース104aはVHDLの予約語かVerilogHDLの予約語か識別するための言語種別301と、VHDLまたはVerilogHDLの予約語302と、予約語302の構文情報（雛形）303から構成される。

【0024】例えば設計者が‘entity’と入力したとする。HDL構文生成部100cは入力された文字列‘entity’を鍵にHDL構文データベース104aの言語種別301VHDL・予約語302‘entity’の検索を行う。検索の結果‘entity’が検出されたら‘entity’構文情報（雛形）303からHDL構文を生成する。

30 【0025】HDL構文データベース104aは予めデータベースとして準備しておく。

【0026】＜図4の説明＞図4は、関連設計データベース104bのデータ構造を示す。

【0027】関連設計データベース104aは、関連記述生成部100dと関連記述修正部100eで使用される。関連設計データベース104aは入力したHDL記述の回路名称401とその回路の端子名称402と端子方向403と端子属性404と関連回路情報405から構成される。回路名称401は回路に対してつけられた名称である。VHDLであれば予約語‘entity’で宣言した回路名称である。またverilogHDLの場合は予約語‘module’で宣言した回路名称である。

40 【0028】端子名称402は回路に付けられた入出力端子の名称である。端子方向403は端子が入力端子（VHDLの場合in、verilogHDLの場合input等）・出力端子（VHDLの場合out、verilogHDLの場合output等）であるか判別するための端子方向の情報である。端子属性404は端子の属性を表わす。たとえばstd_logic型やstd_logic_vector型などの信号の型を表わす。端子名称402・端子方向403・端子属性404は回路の端子本数分生成

する。

【0029】関連設計データベース104bの登録は、回路記述が終了してHDLテキストファイル104cの出力時に回路名称401・端子名称402・端子方向403・端子属性404をHDLテキストファイル104cから抽出し登録処理する。VHDLの場合回路名称は'entity'文から、端子情報は'port'文から作成する。またverilogHDLの場合回路名称は'modul'文から、'input'、'output'、'inout'文から作成する。関連回路情報405は他の関連記述生成部100dで既作成済みHDLテキストファイル104cの回路端子を使用した場合に、参照しているHDLテキストファイル104cの回路名称を登録する。

【0030】図4の場合回路名'ADD'は端子名称402'IN1'・端子方向403'in'・端子属性404'std_logic'の端子を持ち、回路名称401'ADD'は回路名topが参照している事を示している。

【0031】また関連記述修正部100eで端子情報に変更があった場合に関連設計データベース104bの端子名称402・端子方向403・端子属性404も変更

【0032】<図5の説明>図5は、HDL構文生成部100cの原理をあらわす。

【0033】HDLによる回路設計を行う場合、設計者はHDLの文法を学習しなければならず、HDL設計の経験の少ない設計者だとシンタックスミスをおかしやすいという問題がある。

【0034】そこでHDL構文生成部100cは設計者が入力したHDLの予約語を鍵にHDL構文データベース104aの構文情報(雛形)303からHDL構文を生成する。このため設計者はHDLの予約語だけでHDLを記述することができ、HDLの文法を学習する負担を低減できシンタックスミスを防ぐことができる。

【0035】入力画面501は設計者がVHDLで回路のインターフェイス(入出力端子)を規定する予約語'entity'と入力したところである。そして設計者は構文生成指示(例えばキーボードのエスケープキーを押す)を行う。HDL構文生成部100cは文字列'entity'を受け取る。受け取った文字列'entity'を鍵にしてHDL構文データベース104aの言語種別301'VHDL'、予約語302'entity'の検索を行う。検索の結果'entity'が検出されたら構文情報(雛形)303から、'entity'の雛形を生成する。入力画面502は'entity'の雛形が表示されたところを示す。

【0036】設計者は生成された'entity'の雛形構文に回路設計に必要な情報(図5の'?'の部分)を入力し構文を完成させる。HDL構文生成部100cよりHDLの雛形が生成され設計者はシンタックスに気を使わずにすみ経験が少ない設計者でもシンタックスエラーを防ぐ事ができる。

【0037】<図6の説明>HDL構文生成部100cの処理フローを用いてHDL構文生成部100cの処理を詳細に説明する。

【0038】Step600は入力画面から入力された文字列が、HDLの予約語である文字列を鍵にHDL構文データベース104aの言語種別301・予約語302を検索する。入力画面501の場合、言語種別301'VHDL'・予約語302'entity'を鍵にしてHDL構文データベース104aの検索を行う。

10 【0039】Step601はHDL構文データベース104aの検索結果の判定を行う。検索の結果該当する予約語302がHDL構文データベース104aから検出されたならStep602でHDL構文データベース104aの構文情報(雛形)303から文字列(予約語)の雛形を入力画面上に表示する。入力画面502は'entity'の構文情報(雛形)303から'entity'の雛形を表示したことを示す。

20 【0040】またStep601でHDL構文データベース104aの検索結果該当する予約語302が検出されなかった場合には、Step603で該当する予約語(入力文字)が無いと言った趣旨の警告メッセージを出力する。

【0041】Step604は入力画面の入力モードに戻りその後設計者は生成したHDL構文の雛形に対して回路情報(図5の'?'の部分)の入力を行い記述を完成させる。

【0042】<図7の説明>図7は、関連記述生成部100dの原理をあらわす。

30 【0043】HDLで階層化記述の場合に上位階層の回路記述に、下位階層の回路端子情報をもとに端子情報の記述を行う必要がある。HDLの経験がある設計者でも端子情報の記述もれや記述ミスをする可能性がある。関連記述生成部100dは下位階層のHDLテキストファイル104cが作成済みにより関連設計データベース104bに端子情報が既に登録済みならば、関連設計データベース104bの端子情報(端子名称402・端子方向403・端子属性404)とHDL構文データベース104aからHDL構文を生成する。これにより端子の記述漏れや記述ミスを防ぐことができる。

40 【0044】701は設計者が入力画面からVHDLで階層化を表現するの予約語'component'と入力し、その後下位階層の回路名称(この例の場合ADD)を記述し構文生成指示(以下構文生成指示はキーボードのエスケープキーを押すことを意味する)を行う。次に関連設計データベース104bに回路'ADD'が作成済みか回路名称401の検索を行う。検索の結果、下位階層記述'ADD'が登録されていれば、関連設計データベース104b回路名称401'ADD'の端子名称402・端子方向403・端子属性404から'component'文

50 に必要な端子情報を作成する。

【0045】またHDL構文データベース104aから予約語‘component’文の構文情報(雛形)303を検出し雛形をする。関連設計データベース104bから生成した端子情報とHDL構文データベース104aから検出した構文情報(雛形)303をマージして702の入力画面に示す‘component’文を生成する。

【0046】関連設計データベース104bに‘ADD’回路情報がない場合には入力画面703の様に‘component’文の雛形のための生成を行い‘?’の部分に設計者自身で端子情報の記述を行う。

【0047】既に関連設計データベース104bに下位階層の回路情報が登録されている場合関連記述生成部100dによりHDL構文が作成され記述洩れや記述ミスを防ぐことができる。

【0048】<図8の説明>関連記述生成部100dの処理フロー図を用いて関連記述生成部100dの処理を詳細に説明する。

【0049】Step800は関連設計データベース104bの回路名称401を検索する。入力画面701の場合は文字列‘ADD’を鍵に関連設計データベース104bの回路名称401に‘ADD’が登録されているか検索する。

【0050】Step801は検索した結果、回路名称401にデータが登録済みであれば、Step802は関連設計データベース104bの端子情報(端子名称402・端子方向403・端子属性404)から‘component’構文のport文に必要なデータを作成する。入力画面702の場合は関連設計データベース104bの回路名称401‘ADD’の端子情報から‘IN1’, ‘IN2’, ‘S’の端子情報を抽出する。(図4の端子情報は‘IN1’しか記載していないが‘IN2’と‘S’の端子情報が‘IN1’の後ろに繰り返しある)Step803はHDL構文データベース104aから予約語‘component’文の構文情報(雛形)303を抽出し、Step802で作成した端子情報とマージして‘component’文を完成させて入力画面に表示する。入力画面702は‘component’構文の雛形と端子情報(IN1, IN2, S)をマージして生成された‘component’文を示す。

【0051】Step801で関連設計データベース104bに関連記述が無い場合には、Step804で該当する関連情報が無いと言った趣旨の警告メッセージを出力する。

【0052】Step805はHDL構文の雛形を生成するため、HDL構文生成部100cに処理が移り、図6で説明した処理と同じように‘component’文の雛形を生成し入力画面に表示する。

【0053】<図9の説明>図9は、関連記述修正部100eの原理をあらわす。

【0054】論理変更などにより回路の端子記述を変更

する場合がある。その記述変更した回路を端子情報を他のHDLテキストファイル104cで使用している場合変更した端子情報を記述している他のHDLテキストファイル104cの端子記述も同じように変更をしなければならない。この様に変更個所の端子情報を他のHDLテキストファイル104cで使用している場合、他のHDLテキストファイル104cの変更漏れや変更ミスをする恐れがある。

【0055】関連記述修正部100eは関連設計データベース104bから修正個所が他のHDLテキストファイル104cで使用していないか検索し使用していたら端子記述の変更を行う。この関連記述修正部100eにより端子の記述修正により影響がある他のHDLテキストファイル104cの端子の記述修正漏れや修正ミスを防ぐことができる。

【0056】901は回路名称‘ADD’に出力端子‘OUT’を追加修正を示した図である。901aは修正前の記述である。回路の端子は‘IN1’ ‘IN2’ ‘S’の3端子である。901bは901aの記述に出力端子‘OUT’を追加修正し4端子に変更した事を示した図である。902は901の記述変更により回路‘ADD’を参照している他のHDLテキストファイル104cである回路名称‘TOP’の出力端子に‘OUT’追加したことを示す図である。902は回路の階層を表現する記述である。902aは上位階層記述で‘component’文により901aの下位階層回路‘ADD’を用いた構造記述であり‘ADD’の端子情報を共有している。902bは901bで示す様に出力端子‘OUT’を追加したことで‘component’文‘ADD’の‘port’文に出力端子‘OUT’を追加修正した事を示す。

【0057】901bで端子の記述を変更したので関連設計データベース104bの端子情報(端子名称402・端子方向403・端子属性404)を更新する。902の場合関連設計データベース104bの回路名称401‘ADD’レコードに端子情報‘OUT’を追加する。

【0058】次に関連設計データベース104bの回路名称401‘ADD’の関連回路情報405に回路名称があれば、関連回路情報405に登録されている回路名称のHDLテキストファイル104cをHDL入力プログラム100aに読み込み関連する端子記述の変更を行う。

【0059】図9の場合、回路‘TOP’HDLテキストファイル104cの作成時に関連回路情報405に‘TOP’が登録される。回路‘TOP’のHDLテキストファイル104cを読み込み出力端子‘OUT’の追加を行う。

【0060】関連記述修正部100eは既作成済みのHDLテキストファイル104cの変更により他のHDLテキストファイル104cの修正が必要な関連設計データベース104bから検索し、修正が必要なHDLテキストファイル104cがあれば修正を行う。これにより変更もれや変更ミスを防ぐことができる。

【0061】<図10の説明>関連記述修正部100eの処理フロー図を用いて関連記述修正部100eの処理を詳細に説明する。

【0062】Step1000は設計変更等により回路の入出力端子記述を変更したので、変更中の回路名称を鍵に関連設計データベース104bの回路名称401を検索する。図9の場合は回路名称401‘ADD’を鍵に関連設計データベース104bの回路名称401を検索する。

【0063】Step1010は関連設計データベース104bの回路名称401‘ADD’レコードに端子情報‘OUT’の追加処理を行いデータを更新する。図9の場合、端子名称402‘OUT’・端子方向403‘out’・端子属性404‘std_logic’が追加される。

【0064】Step1020は関連設計データベース104bの関連回路情報405を検索し関連回路情報405に回路名称が登録されているか検索する。図9の場合は関連回路情報405に回路名‘TOP’が登録されている。(関連回路情報405の登録は回路名称‘TOP’のHDLテキストファイル104cの作成時に‘component’文の‘ADD’を記述する際に関連記述生成部100dで登録される) 関連回路情報405に回路名称が登録されていればその回路名称のHDLテキストファイル104cをHDL入力プログラム100aに読み込み、Step1010で更新された端子情報(端子名称402・端子方向403・端子属性404)から変更データを作成して端子記述の変更を行う。図9の902bの場合は、端子情報‘OUT’が追加される。

【0065】Step1040はStep1020の関連設計データベース104bの検索の結果、関連回路情報405に回路名称が登録されていない場合は通常の入力モードに戻り設計者がHDL入力プログラム100aの入力画面から入力を行う。

【0066】<図11の説明>図11は、セマンティックスチェック部100fの原理をあらわす。

【0067】回路設計では設計した回路が正しく動作するかシミュレータを用いてシミュレーションを行う。シミュレーションではVHDLの‘process’文、VerilogHDLの‘always’文に含まれるセンシティブティリストの信号の変化を検証する。そのため信号代入文の右辺の信号、または条件式内の信号がセンシティブティリストに記述する必要がある。(ただしVHDLの場合‘wait’文の記述が‘process’文内にある場合は記述不可) HDL記述の初心者でなくてもセンシティブティリストの記述もれをする恐れがあり回路の検証工程で問題となる。セマンティックスチェック部100fはセンシティブティリストに必要な信号を検出してセンシティブティリストを作成する。これにより不完全なセンシティブティリストの作り込みを防ぎ回路の検証工程での問題を防ぐことが出来る。

【0068】1101様にセンシティブティリストの記述にもれがある場合、1102のように信号代入文の右辺または条件式内の信号はセンシティブティリストに追加をする。(センシティブティリストとは‘process’文または、‘always’文の後ろの‘()’内の事である) 図11の場合1101には‘B’‘SEL’がセンシティブティリストに記述されていないので、1102で示す様に‘B’‘SEL’をセンシティブティリストに追加する。

【0069】図11の1103は未使用な信号・端子のチェックを行う。1103では‘signal’宣言で‘A__SIG’と記述されているが機能記述部(process文内)で使用されていない。このような記述を残しておく不要な端子を生成し回路の面積が大きくなったりして回路の品質が悪くなる恐れがある。このような未使用な信号・端子が存在した場合警告メッセージを出力する。設計者は警告メッセージにより記述に不要な端子や信号がないか確認がとれ回路の品質を低下することを防ぐことができる。

【0070】<図12の説明>セマンティックスチェック部100fの処理フローを用いてセマンティックスチェック部100fの処理を詳細に説明する。

【0071】Step1200はHDL構文生成部100cの処理により‘process’文や‘always’文の生成を行う。

【0072】Step1210は生成された‘process’文や‘always’文にHDL入力プログラム100aの入力画面から設計者が回路の機能記述を行う。この時センシティブティリスト内の信号情報をワークエリア100gにテーブルA、条件式や信号代入文の右辺の信号情報はワークエリア100gにテーブルBとして記憶しておく。

【0073】Step1220は機能記述が終了したら、センシティブティリスト内の信号情報(テーブルA)と、条件式や信号代入文の右辺の信号情報(テーブルB)を比較してセンシティブティリストに記述もれがあればテーブルBからデータを作成してセンシティブティリストに追加する。

【0074】図11の1101の場合ワークエリア100gのテーブルAには‘A’が、ワークエリア100gのテーブルBには‘A’‘B’‘SEL’を記憶する。テーブルAとテーブルBの比較の結果テーブルAに記憶されていない‘B’‘SEL’をセンシティブティリストに追加する。

【0075】Step1230は信号宣言または入出力端子情報を信号・端子情報データとしてワークエリア100gのテーブルCに記憶する。Step1240は機能記述部の信号情報を信号情報データとしてワークエリア100gのテーブルDに記憶する。

【0076】Step1250は、信号・端子情報デー

タ（テーブルC）と信号情報データ（テーブルD）を比較する。Step 1250はテーブルCとテーブルDを比較した結果登録データに過不足があった場合、過不足となった信号・端子情報について警告メッセージの出力を行う。

【0077】図11の1103の場合信号・端子情報データ（テーブルC）には‘port’情報‘I1’‘I2’‘SEL’‘01’と、signal情報‘A_SIG’が記憶される。信号情報データ（テーブルD）には機能記述部（process文内）の‘I1’‘I2’‘SEL’‘01’が記憶される。テーブルCとテーブルDの比較結果‘A_SIG’が信号宣言しているが未使用な信号と判断され‘A_SIG’に対して警告メッセージを出力する。

【0078】

【発明の効果】以上述べた如く本発明を用いれば、HDLを用いて論理設計を行う上で、前記従来手法の問題点を解決し、HDL記述を用いて設計を行う事ができる。

【図面の簡単な説明】

【図1】HDL入力システムのシステム構成図である。

【図2】HDL入力システムのプログラム構成図である。

【図3】HDL構文データベースのデータ構造をあらわす。

【図4】関連設計データベースのデータ構造をあらわす。

【図5】HDL構文生成部の原理をあらわす。

【図6】HDL構文生成部の処理フロー図である。

【図7】関連記述生成部の原理をあらわす。

【図8】関連記述生成部の処理フロー図である。

【図9】関連記述修正部の原理をあらわす。

【図10】関連記述修正部の処理フロー図である。

【図11】セマンティックチェック部の原理をあらわす。

【図12】セマンティックチェック部の処理フロー図である。

【符号の説明】

100…主記憶装置、

100a…HDL入力システム、

100b…HDL入出力部、

100c…HDL構文生成部、

100d…関連記述生成部、

100e…関連記述修正部、

100f…セマンティックチェック部、

101…CPU、

102…ディスプレイ、

103…キーボード、

104…ハードディスク、

104a…HDL構文データベース、

104b…関連設計データベース、

104c…HDLテキストファイル、

105…バス、

301…言語種別、

302…予約語、

303…構文情報（雛形）、

401…回路名称、

402…端子名称、

403…端子方向、

20 404…端子属性、

405…関連回路情報、

501…初期入力画面、

502…HDL構文生成画面、

701…初期入力画面、

702…自動生成画面（関連情報あり）、

703…自動生成画面（関連情報なし）、

901…記述変更側入力画面、

901a…記述変更前入力画面、

901b…記述変更後入力画面、

30 902…関連記述側入力画面、

902a…自動変更前入力画面、

902b…自動変更後入力画面、

1101…自動生成前入力画面、

1102…自動生成後入力画面、

1103…セマンティックチェックの記述例。

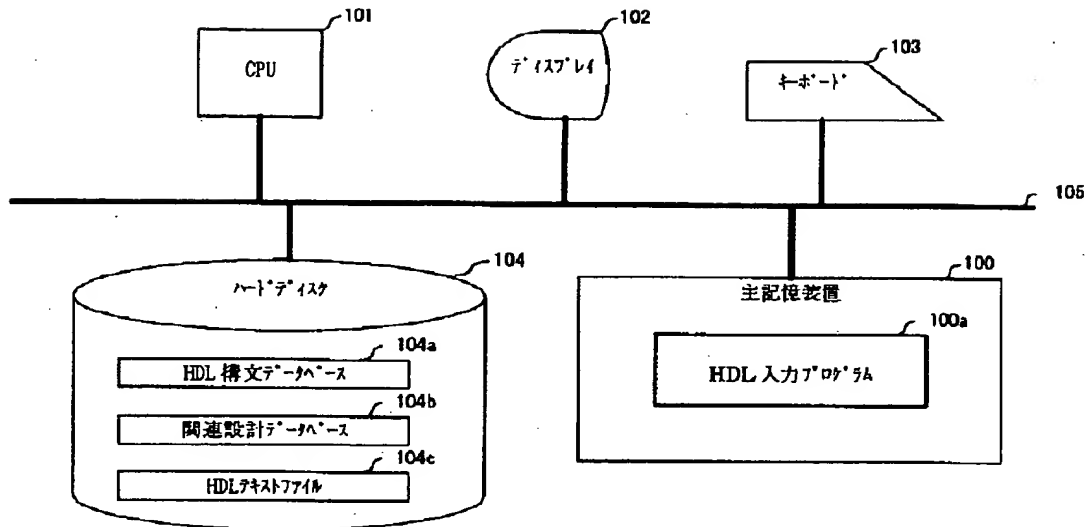
【図3】

図3

301 言語種別	302 予約語	303 構文情報（雛形）	104a
VHDL	entity	entity ??? is generic(??? in ???=???); port(???in ???; ??? out ???); end ???;	
Verilog	module	module ??? (???); endmodule	
:	:	:	

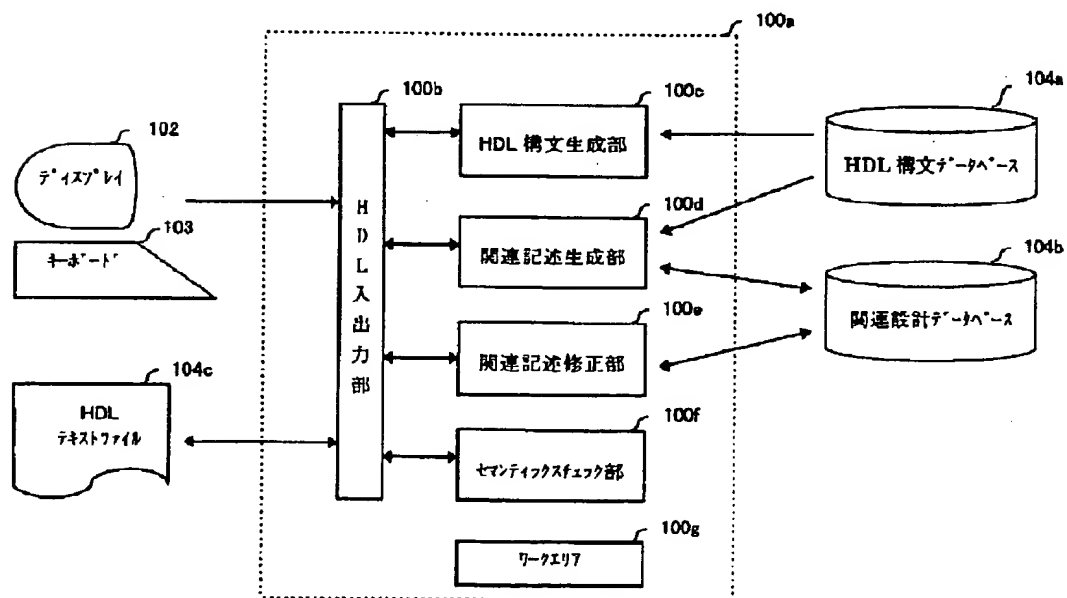
【図 1】

図 1



【図 2】

図 2



【図 4】

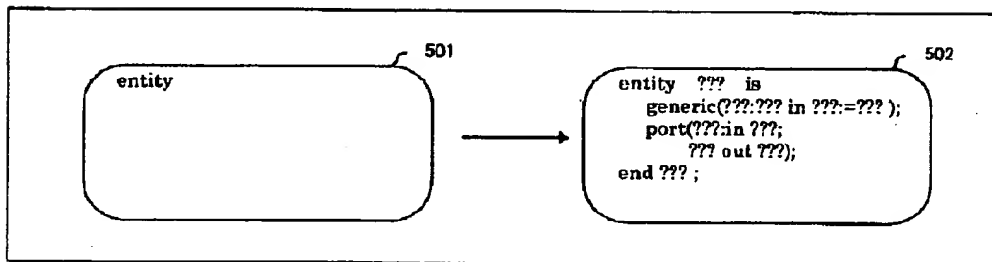
図 4

401 回路名称	402 端子名称	403 端子方向	404 端子属性	...	405 関連回路情報
ADD	IN1	in	std_logic		top
:	:	:	:	:	:

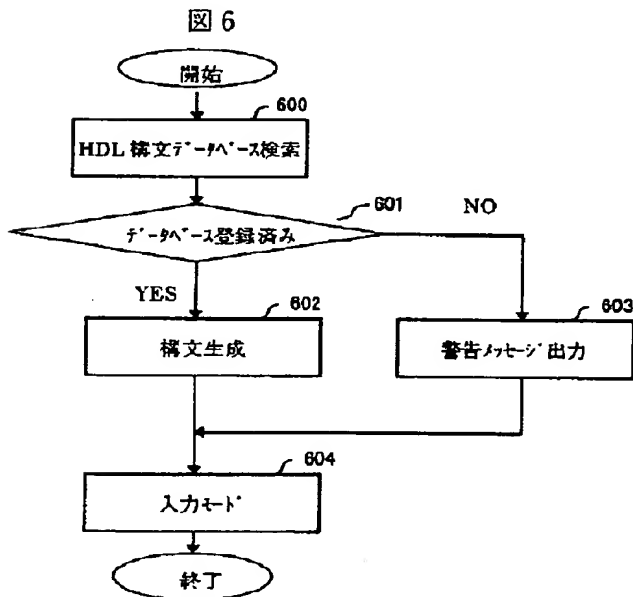
端子数分 402・403・404 の繰り返し

【図 5】

図 5

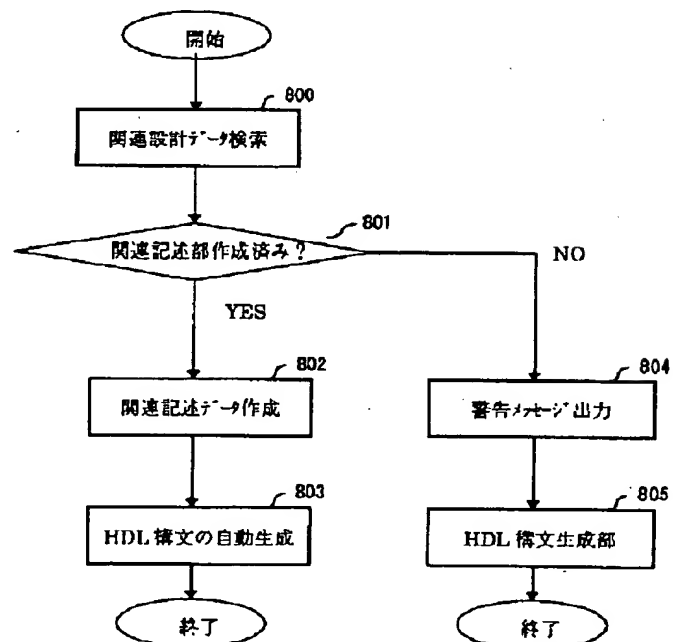


【図 6】



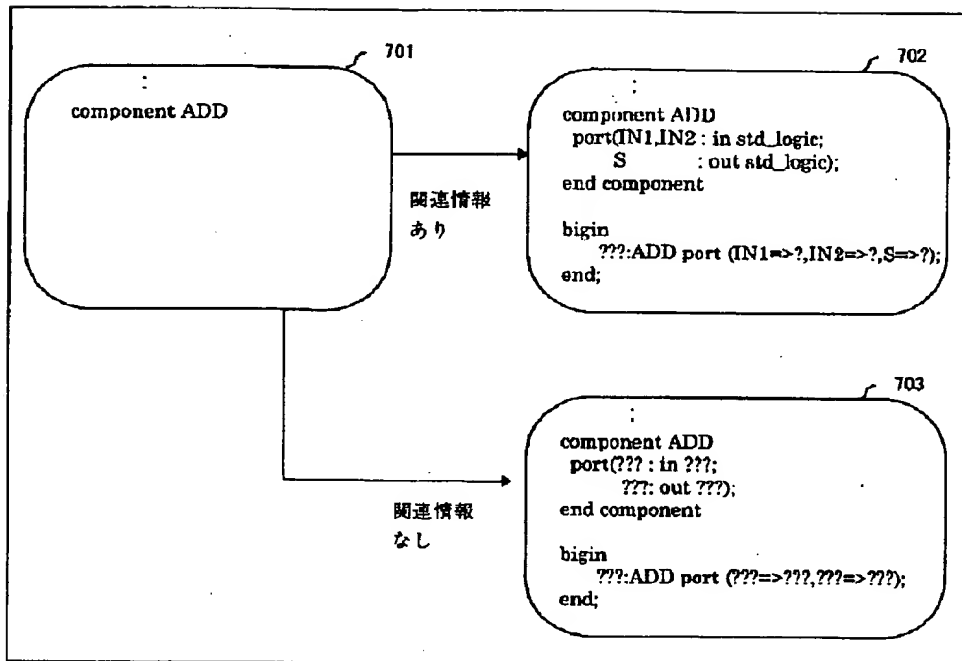
【図 8】

図 8



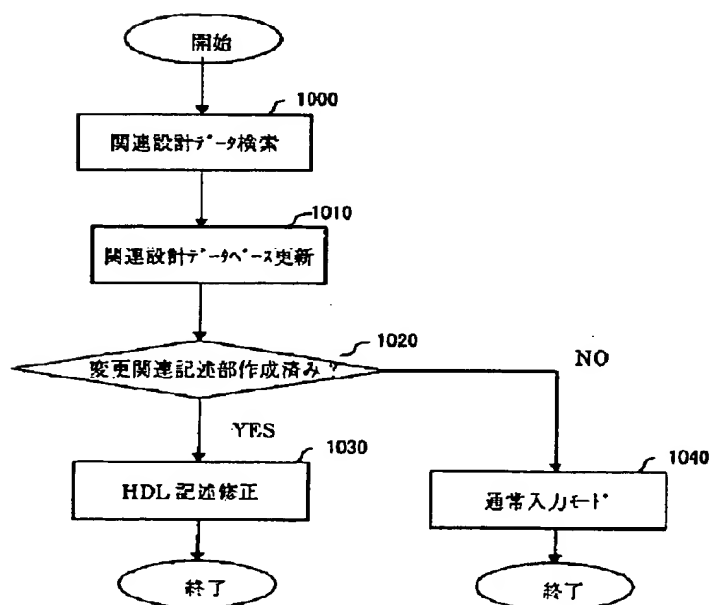
【図7】

図7



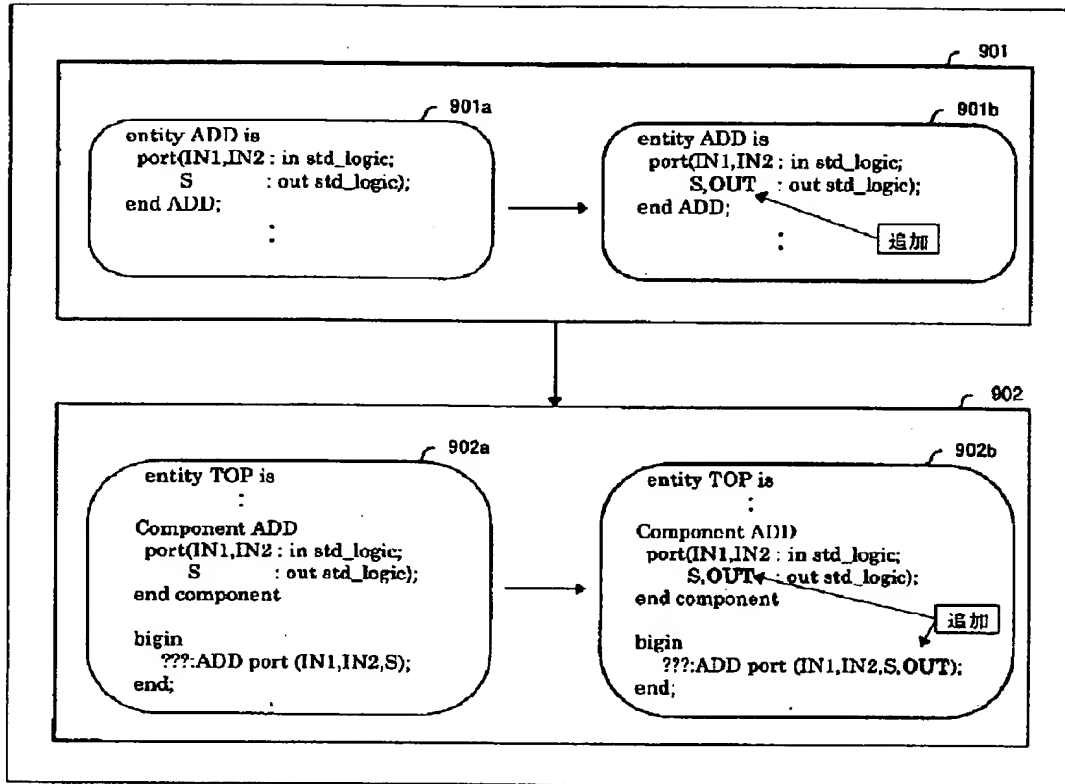
【図10】

図10



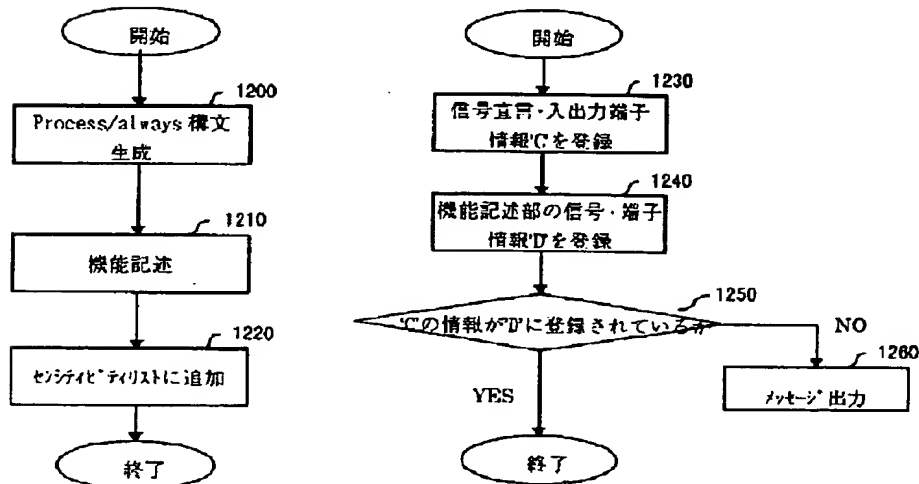
【図 9】

図 9



【図 12】

図 12



【図 11】

図 11

